A Hybrid Method of Assurance Cases and Testing for Improved Confidence in Autonomous Space Systems

Ben Smith, Martin S. Feather and Terry Huntsberger
Jet Propulsion Laboratory, California Institute of Technology

Jet Propulsion Laboratory
California Institute of Technology

Acknowledgements

This research was carried out at the Jet Propulsion Laboratory, California Institute of Technology under a contract with the National Aeronautics and Space Administration and funded through the internal Research and Technology Development program.

The authors thank Mike McHenry for providing us plentiful information on M2020, and Ewen Denny for allowing us use of AdvoCATE for creating and editing assurance cases.

Autonomy for space exploration

- Autonomous systems react intelligently to their environments
  - Capable of handling many possible conditions
  - For us, useful for space exploration
- BUT: How do we develop confidence they will do the right thing?
  - Challenging to test those many conditions
  - For us, limited opportunities for testing

Assuring autonomous systems – our approach

Combine two existing methods:

- **Assurance Case**: a rigorous argument that the system satisfies a vital property
  - E.g., the Mars rover will remain safe while moving (not crash, roll over, get stuck)
  - Use it to derive the set of conditions to test
- **High Throughput Testing (HTT)**
  - Generate the minimal test suite needed to provide a desired level of test coverage

Assurance (Safety) Cases – e.g., Tim Kelly, Univ. York, UK

HTT (Combinatorial Testing) – Kuhn et al, NIST

Assurance Cases

"An assurance case is an organized argument that a system is acceptable for its intended use with respect to specified concerns (such as safety, security, correctness)." [Reinhart, Knight and Rowanhill]

A generalization of "safety cases" used widely in Europe.

- Decomposes a claim into sub-claims
- Ultimately leads to evidence – tests, analyses, historical information, etc.

High Throughput Testing

For a desired coverage level, minimizes the number of test cases needed. Example:

A	B	C	D	E	F	G	H	I	J
0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1	1
1	1	1	0	1	0	0	0	0	1
1	0	1	1	0	1	0	1	0	0
1	0	0	0	1	1	1	0	0	0
0	1	1	0	0	1	0	0	1	0
0	0	1	0	1	0	1	1	1	0
1	1	0	1	0	0	1	0	1	0
0	0	0	1	1	1	0	0	1	1
0	0	1	1	0	0	1	0	0	1
0	1	0	1	1	0	0	1	0	0
1	0	0	0	0	0	0	1	1	1
0	1	0	0	0	1	1	1	0	1

10 binary-valued variables

13 test cases

36 triples (ABC, ABD, … HIJ). For *every* triple its eight value combinations (000, 001, … 111) are present! E.g.:

ABC's eight combinations

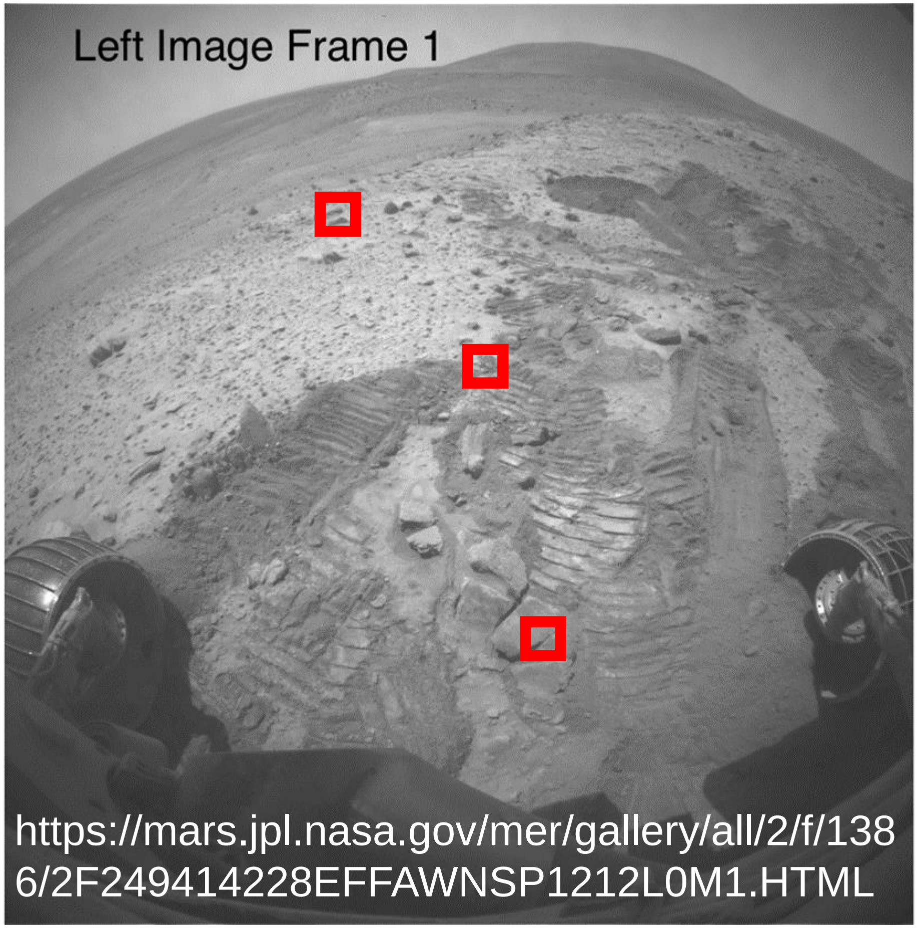
DEG's eight combinations

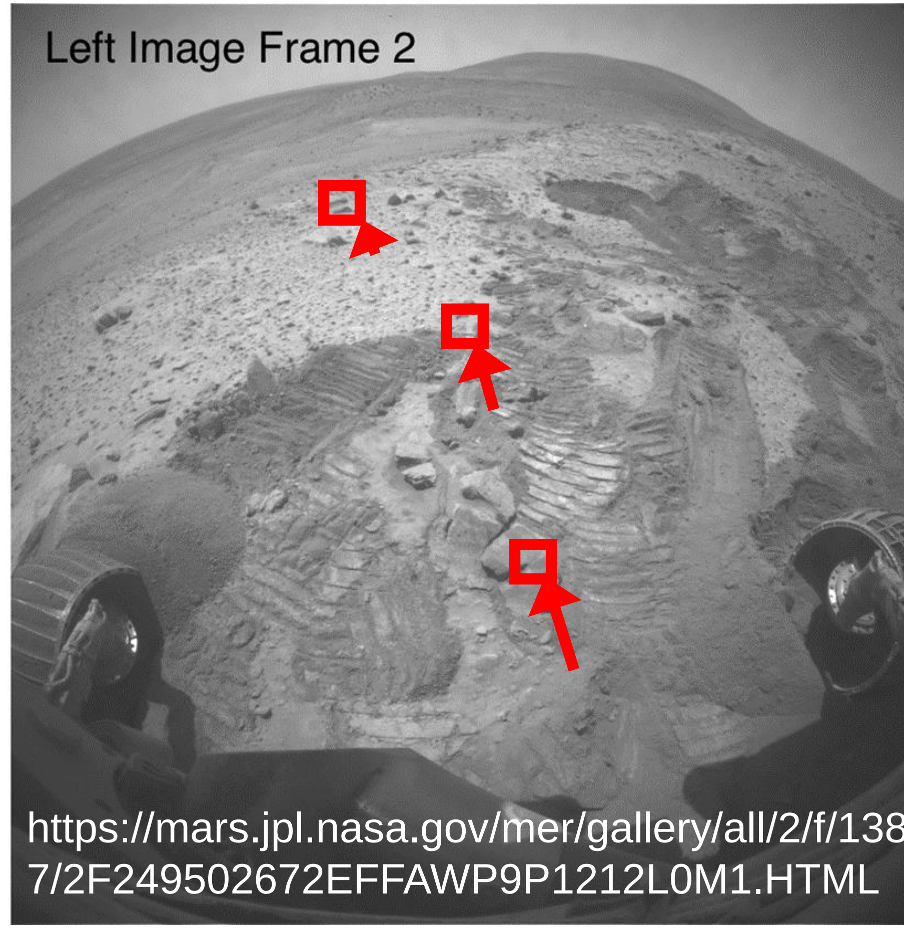
HIJ's eight combinations

A Mars rover example

- Over-the-horizon *autonomous* driving
  - Plans and executes a safe route towards destination
  - Allows the rover to drive further each (Martian) day
- Our Assurance Case for its safety identifies what to test and why – for example:
- After each move segment the rover re-determines where it is – so as to avoid forbidden regions as it continues
  - While driving, its wheels may slip, so cannot determine distance or direction from wheel turns
  - There’s no GPS on Mars!
  - “Visual Odometry” used instead (*see next slide*) and therefore is crucial to test

Visual Odometry (a simplified explanation)





Select features in the before-move image

Look for them in the after-move image

From movement of features between images, and knowledge of distances (from stereo images), deduce the rover's movement:

Visual Odometry – what could go wrong?

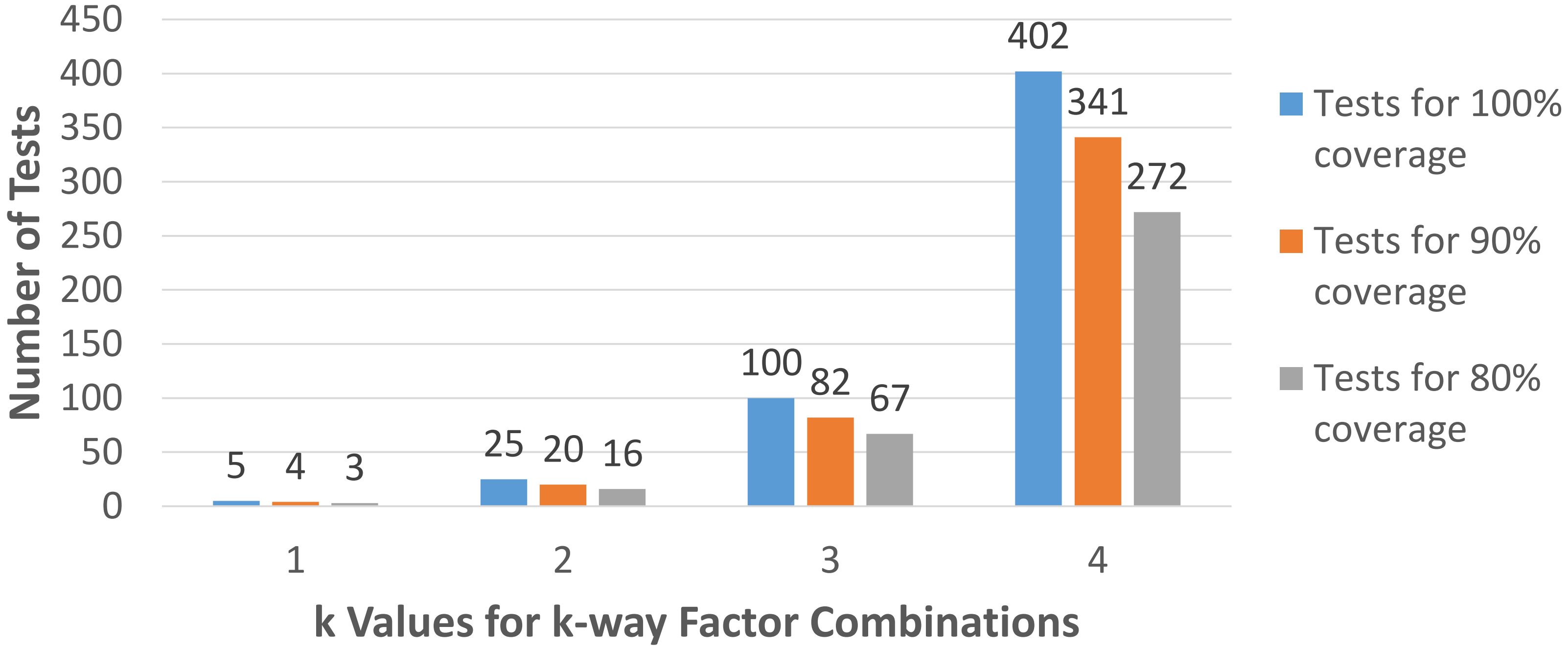
Hazard		Factor	Value(s)
Incorrectly matching features	Repetitive terrain	Terrain texture	Rippled
	Featureless terrain	Terrain texture	Smooth
	Challenging lighting	Lighting	Low, High, Vertical
	Degraded optics	Optics	Degraded
	Little/no image overlap	Ground Interaction	Low & High Slip, Low & High Skid
		Motion	Long Distance, Large Rotation
Non-terrain features	Rover's own shadow	Lighting	Self shadow
	Dust on lens	Optics	Degraded
	Failed camera pixels	Optics	Degraded
Insufficient accuracy	Little parallax	Distant features	Present
	Poor camera resolution	Optics	Degraded

Visual Odometry – what to test?

Factor	Values (union of nominal and hazard)		#
	Nominal	**Hazard**	
Distant features	Absent	Present	2
Ground interaction	Nominal	Low & High Slip, Low & High Skid	5
Lighting	Nominal	Low, High, Vertical, Self shadow	5
Motion	Short distance, Small rotation	Long distance, Large rotation	4
Optics	Nominal	Degraded	2
Terrain texture	Medium, High	Smooth, Rippled	4

2 x 5 x 5 x 4 x 2 x 4 = 1,600 combinations of all six factors' values

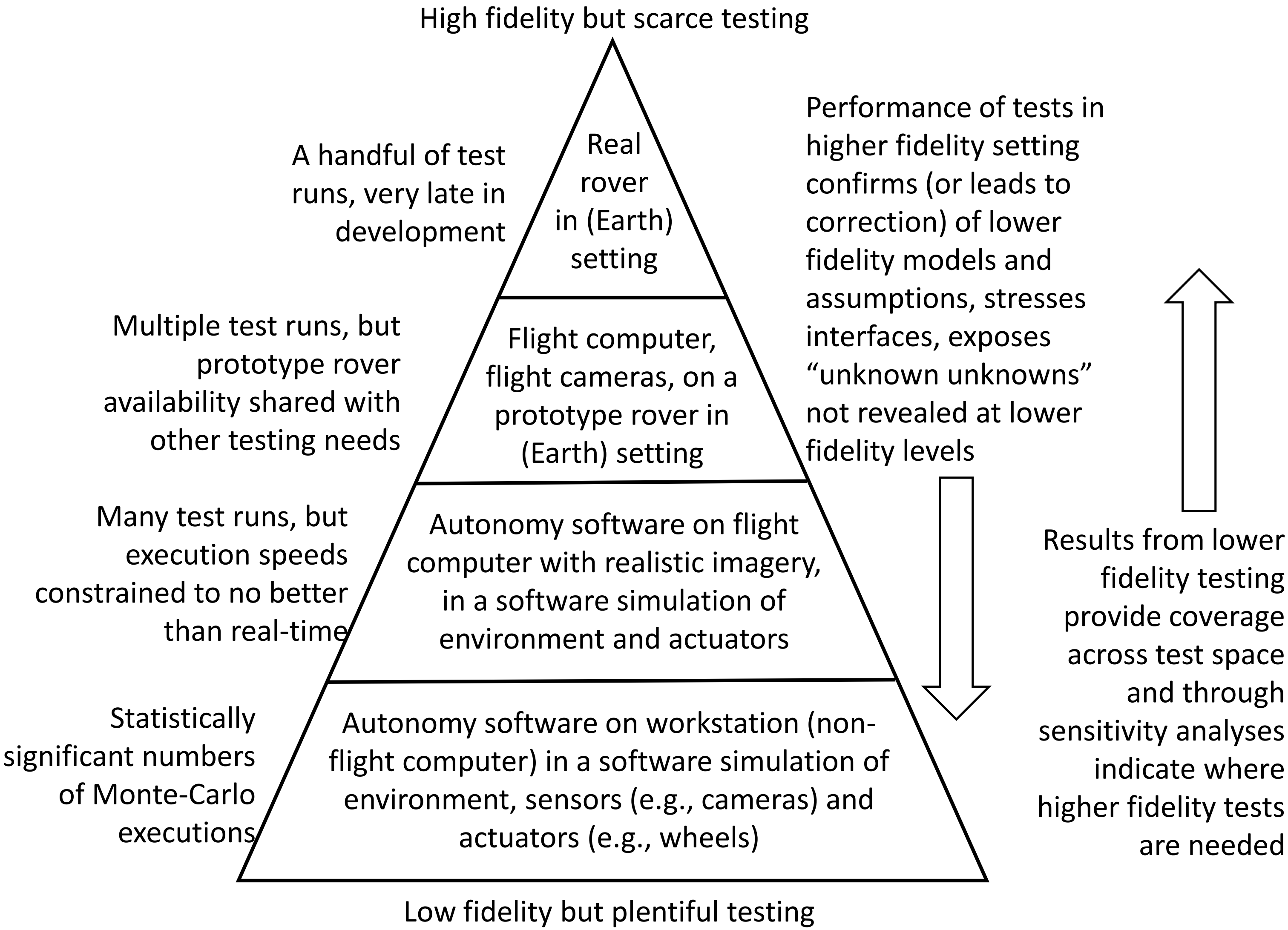
Test suite generation using HTT



Summary

- Autonomy desired because it can handle many possible conditions – but *testing* those many conditions is challenging
- Derive the critical tests needed to have confidence in the autonomy:
  - Assurance case used to derive the conditions to test
  - High Throughput Testing used to generate the minimal test suite needed to provide a desired level of test coverage

Levels of test venue fidelity



A question for YOU!

- How to determine what tests to run at what levels of fidelity?
  - Past work on this?
  - Your ideas?
  - Your interest in this area?

Jet Propulsion Laboratory
California Institute of Technology

jpl.nasa.gov